



JavaScript em Páginas Web



lá Padawan!

Nesta unidade veremos como vamos conectar de forma programática nossos documentos HTML com o JavaScript, para que possamos criar interações dinâmicas e tudo mais que nossos jogos web precisam. Tem muita coisa boa para aprendermos!

Vamos estudar sobre o *Document Object Model* e como esse modelo nos permite acessar elementos e criar eventos em aplicações e jogos de páginas web.

Como funciona o *Document Object Model* (DOM)

Quando o seu navegador carrega uma página web, ele cria o modelo de objetos da página, para que seja possível acessá-los de forma programática. Chamaremos isso de DOM - *Document Object Model*. Por meio desse recurso temos uma interface de programação para usar a linguagem JavaScript e tratar o HTML, bem como os estilos CSS, de forma dinâmica.

O modelo DOM define:

- Os elementos HTML como objetos (veremos mais sobre objetos nas próximas unidades);
- As propriedades desses objetos com base nos elementos HTML;

- Métodos para acessar os elementos HTML de uma página web;



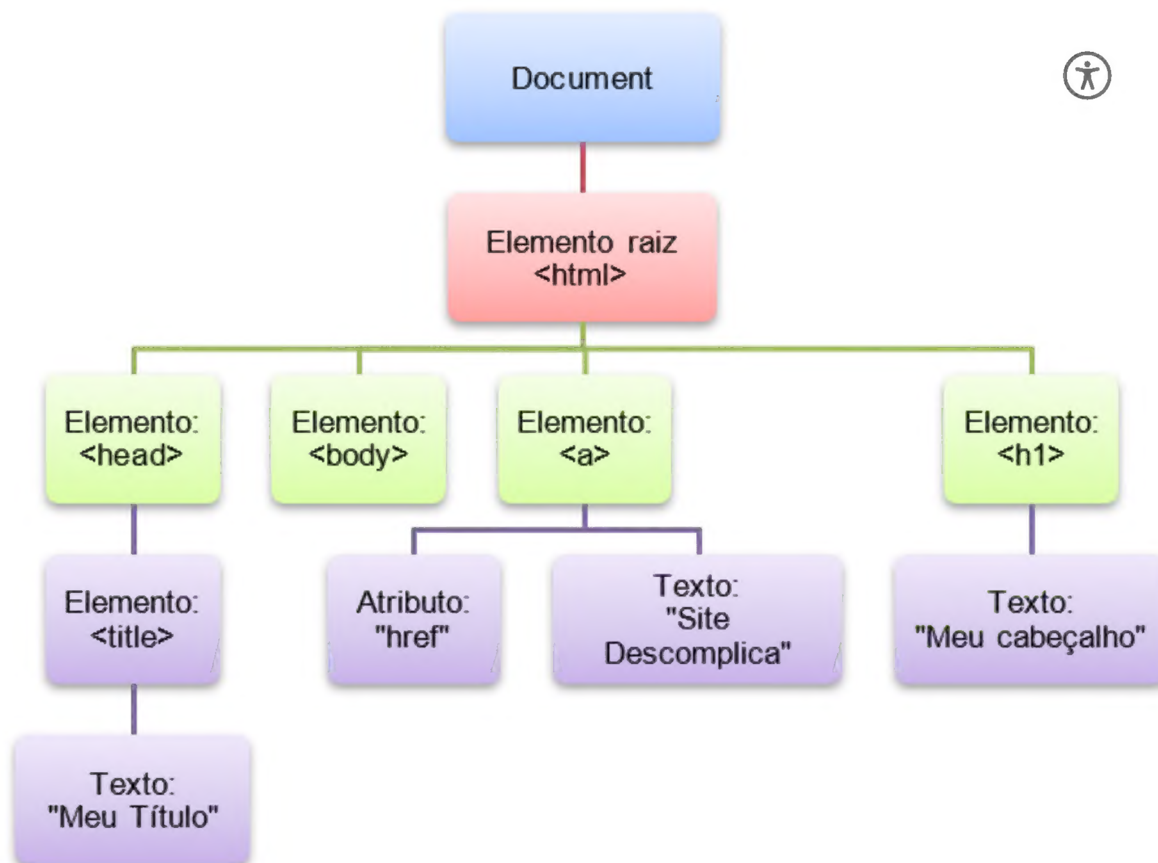
- Uma série de eventos para interagirmos com os elementos HTML.

Assim, existe um padrão de como obter, alterar, adicionar ou excluir elementos HTML de uma aplicação web qualquer.

O modelo DOM constrói um documento HTML como uma árvore de objetos. Essa árvore define o caminho para que possamos acessar os objetos de um documento HTML e suas propriedades. Vejamos como exemplo o documento HTML a seguir:

```
<!DOCTYPE html>
<html>
    <head>
        <title>Meu Título</title>
    </head>
    <body>
        <h1>Meu cabeçalho</h1>
        <a href="descomplica.com">Site Descomplica.</a>
    </body>
</html>
```

A árvore DOM deste documento HTML de exemplo seria como a figura a seguir:



Então, através do JavaScript, podemos acessar os elementos deste documento e modificá-los dinamicamente. Poderíamos alterar o título da página, por exemplo, com o seguinte trecho de código:

document.title = ‘Link para o site descomplica’;

Com isso o texto dentro do elemento <title> agora foi alterado de “Meu título” para “Link para o site descomplica”.

Isso tudo será essencial para nossos jogos web e principalmente para a interação e interfaces de nossos jogos. Pois agora podemos:

- Alterar todos os elementos HTML, bem como seus atributos;
- Podemos mudar todos os estilos CSS na página;
- Criar ou remover elementos e atributos em tempo de execução HTML;

- Reagir a eventos HTML e ações do usuário na aplicação web.



Com isso, o DOM nos fornece na sua interface vários métodos e propriedades. Um método é uma ação que você pode realizar (como adicionar ou excluir um elemento na página web). Já uma propriedade é um valor de um elemento HTML que você pode acessar, inserir ou alterar (como alterar a propriedade href de um elemento <img>).

Para que possamos manipular elementos HTML com JavaScript, devemos acessar estes elementos primeiro, devemos buscá-los para vincular ao nosso programa. Algumas maneiras de fazer isso são: buscar elementos com base na sua propriedade id; encontrá-los pelo nome de tags, nome de classes ou através de seletores CSS.

Para encontrar um elemento pela sua **propriedade id**, utilizaremos a seguinte sintaxe:

document.getElementById(*id*)
Encontrar elementos em função do **nome da tag**, faremos:
document.getElementsByTagName(*nome_tag*)
Para encontrar elementos em função do **nome da classe CSS**, faremos:
document.getElementsByClassName(*nome_classe*)

Tanto o método getElementsByTagName(), quanto getElementsByClassName() retornam uma coleção de objetos, ou seja, um array. Observe que o nome do método está no plural.

O método getElementById

A maneira mais direta para acessar um elemento HTML específico é buscá-lo por meio de sua propriedade id. Como essa propriedade de um identificador único para um elemento, poderemos acessá-lo diretamente. O método getElementById() será muito utilizado em nossos games.

Como exemplo, se eu tivesse os seguintes elementos numa página HTML:

```
<a id="tipo-arma" href="/tipo/espada-curta">Espada curta</a>
<p id="descricao-arma">Espada de dois gumes com 60cm.</p>
```

Poderíamos acessá-los e armazenar seus objetos em variáveis JavaScript para tratamentos futuros da seguinte forma:

```
// Armazenar o objeto da tag <a> com id = tipo-arma
let tipo = document.getElementById("tipo-arma");
// Armazenar o objeto da tag <p> com id = descricao-arma
```

```
let descricao = document.getElementById("descricao-arma");
```

Tratamento de Eventos

Em qualquer linguagem de programação, os eventos são ações ou ocorrências que acontecem na aplicação que estamos desenvolvendo. O sistema irá disparar algum tipo de sinal quando o evento acontecer, provendo um mecanismo para que blocos de código específicos sejam executados.

No nosso caso, esse mecanismo é feito com métodos definidos pelo DOM. Então, em aplicações web no lado do cliente, estes eventos são disparados geralmente dentro do navegador e estão vinculados aos elementos HTML.

Podemos definir eventos para um único elemento, um conjunto de elementos ou mesmo eventos relacionados à janela do navegador.

Eles podem representar interações básicas do usuário como cliques do mouse, teclas pressionadas ou rolagem da página. Veja alguns eventos úteis para nossos jogos:

- O usuário clicando com o mouse ou passando o cursor do mouse sobre um certo elemento;
- O usuário pressionando teclas específicas do teclado;
- O usuário redimensionando ou fechando a janela do navegador;
- Um vídeo sendo reproduzido, pausado ou terminando sua reprodução;
- Detecção de erros e alterações diretas no DOM.

Para definirmos um comportamento específico para o nosso programa por intermédio dos eventos, precisamos registrar um manipulador de eventos (do inglês *event handler*). Dentro do *event handler*, vamos inserir nosso bloco de código que deve ser executado quando o evento que definimos for disparado.

Para registrar manipuladores de eventos em JavaScript usaremos o método addEventListener(). Sua sintaxe é a seguinte:

```
function funcManipuladora() {
  // Bloco de código a ser executado quando o evento é disparado
}
```

```
// Registrando a função contendo o manipulador de eventos
objeto.addEventListener('tipoEvento', funcManipuladora);
```



Note que o método possui dois argumentos: o primeiro especifica qual o tipo do evento e o segundo faz referência à função que contém o bloco de código propriamente dito, que será executado quando o evento for disparado.

Para o argumento de tipo de evento, existem várias possibilidades. Vejamos algumas que serão bem comuns em nossos games:

Evento	Descrição
onchange	Executa o evento após um elemento HTML ser alterado.
onclick	O evento dispara após o usuário clicar em um elemento.
onmouseover	Quando o usuário move o mouse sobre um elemento.
onmouseout	Quando o usuário move o mouse para longe de um elemento.
onkeydown	Quando uma tecla do teclado é pressionada.
onload	Dispara o evento após terminar de carregar a página.

Vamos a um exemplo? Se tivéssemos os seguintes elemento </button> numa página HTML:

```
<button id="btn-exemplo">Clique-me</button>
```

Poderíamos criar um programa JavaScript que, quando o usuário clique no botão, sua cor de fundo seja alterada para vermelho. Da seguinte forma:

```
let btn = document.getElementById('btn-exemplo');
function trocarCor() {
  let bkgColor = btn.style.backgroundColor;
  btn.style.backgroundColor = (bkgColor === 'red') ? 'lightgrey' : 'red';
}
```



```
btn.addEventListener('onclick', trocarCor);
```

Chegamos ao fim na nossa primeira metade das unidades! Tem sido um aprendizado muito intenso, não é? Mas todos estes assuntos que temos visto possuem uma documentação muito ampla para estudos em livros e na internet. Por isso não deixe de conferir o que deixei para você em atividade extra!

Abraços!

Atividade Extra

Os sites do Mozilla e da W3 Schools possuem uma referência completa sobre eventos em JavaScript principalmente e o Document Object Model. Na aula não é possível ver todas as possibilidades de tipos de eventos, manipuladores e atributos DOM, pois esse conteúdo é muito extenso. Por isso, você tem como tarefa extra, acessar os seguintes sites para se aprofundar:

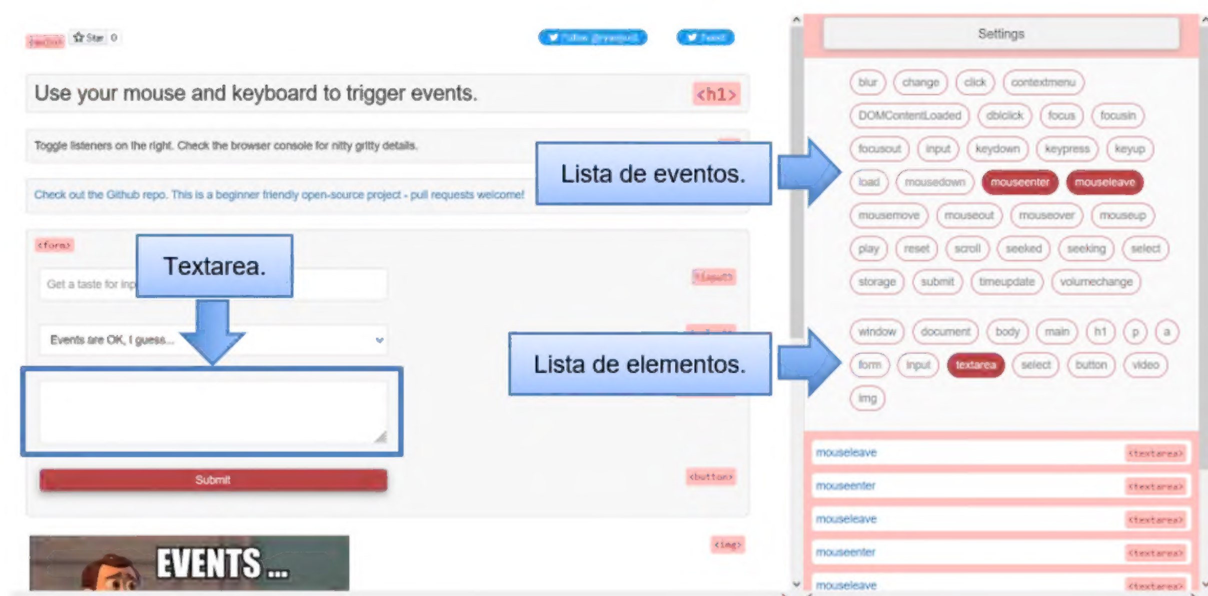
- **MDN Web Docs JavaScript Events Reference.** Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/Events>.

- **HTML DOM Events.** Disponível em: <https://www.w3schools.com/jsref/dom_obj_event.asp>.



DOM Events Playground

Acesse <https://ryanjyost.github.io/dom-events**>** e experimente vários tipos de eventos. Nesse site você pode selecionar o tipo de evento, bem como a qual objeto o evento será registrado. Por exemplo, selecione na lista de eventos **mouseenter** e **mouseleave**. Em seguida escolha o elemento HTML **textarea** disponível na lista de elementos. Veja o exemplo na imagem:



Agora mova o mouse sobre o elemento **textarea** à esquerda da tela e observe o que acontece: os eventos forem detectados, eles serão listados na área direita.

Referência Bibliográfica

MDN WEB DOCS. **Introdução a eventos.** Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Building_blocks/Events>.

W3 Schools. **JavaScript Html Dom**. Disponível em: <https://www.w3schools.com/js/js_htmldom.asp>.

Ir para exercício